Project 1——向 Linux 内核添加系统调用

5142029014 高超

一、实验内容

学习 Linux 操作系统提供的系统调用接口，以及一个用户程序如何通过该接口与操作系统内核实现通信。任务是向 Linux 内核中添加一个系统调用，扩展该操作系统的功能。

二、实验环境

虚拟机上安装 Ubuntu 15.04 64 位。

从 http://www.kernel.org/上下载的内核 linux-3.12.59。

二、对于实验的理解与基本思路

系统调用是操作系统提供的完成特定功能的函数。关于系统调用的代码实现在/usr/src/linux-3.12.59/kernel/sys.c 文件下。我决定使用比较简便的做法选择直接在源文件下添加自己的程序，这样会比较方便，不需要修改 Makefile。

用户在调用系统调用时会向内核传递一个系统调用号，接着系统调用处理程序能通过此号从系统调用表中找到对应的内核函数执行。所以还需要为我自己的系统调用函数申请一个系统调用号，同时要添加函数声明。

如上修改完三处地方以后，就可以开始编译内核了，编译成功后进行安装。接下来启动进入新的内核以后需要编写用户态函数调用添加的系统调用来验证是否添加成功。

三、实验的具体实现

1.一些准备

下载内核 linux-3.12.59，解压至/usr/src 目录

下载源码配置的工具

sudo apt-get install libncurses5-dev

2.内核代码的修改

（1）在/usr/src/linux-3.12.59/kernel/sys.c 文件中

添加头文件

```
#include <linux/linkage.h>
```

在源代码最后加上自己的程序（传递一个参数）

```
asmlinkage int sys_hello(int i)
{
    printk(KERN_EMERG "          this is the project %d\n", i);
    printk(KERN_EMERG "          hello myKernel!\n");

    return 1;
}
```

（2）/usr/src/linux-3.12.59/arch/x86/syscalls/syscall_64.tbl

申请一个系统调用号(32 位系统修改 syscalls_32.tbl)

```
311 64  process_vm_writev   sys_process_vm_writev
312 common  kcmp            sys_kcmp
313 common  finit_module    sys_finit_module
314 64      hello           sys_hello
```

314 为添加的系统调用号

（3）/usr/src/linux-3.12.59/include/linux/syscalls.h

最后添加函数声明

```
asmlinkage int sys_hello(int i);
```

3.内核的配置，编译和安装

Make menuconfig　　直接选择默认配置即可

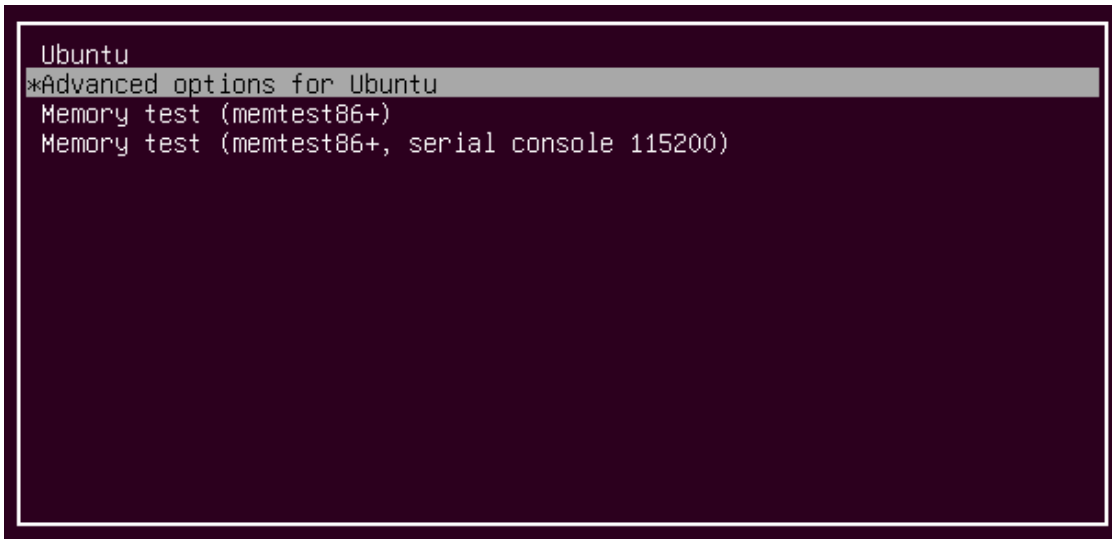
Make –j4　　多核编译

Make modules_install　安装模块

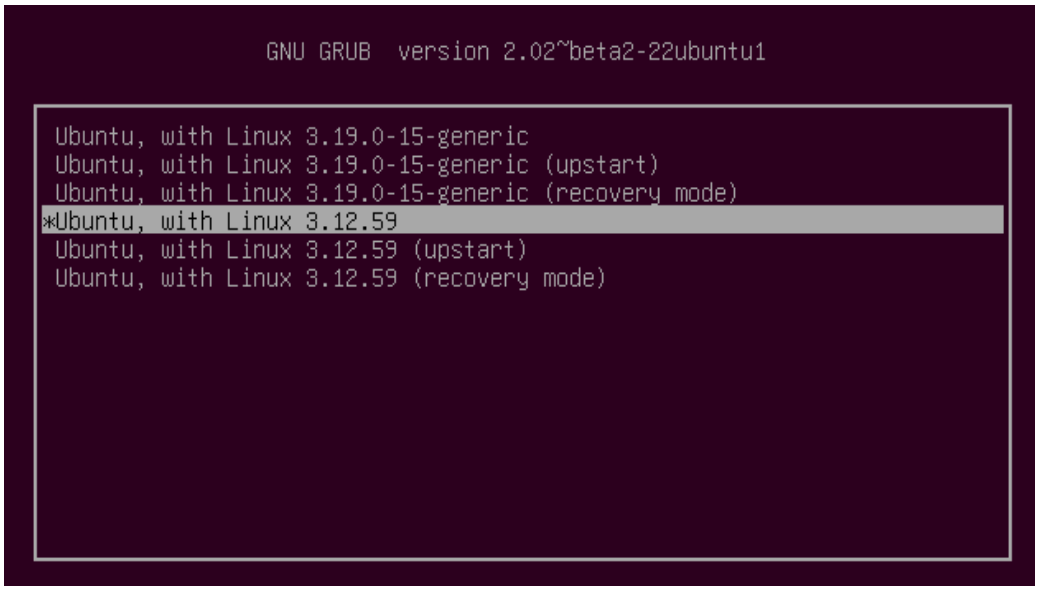
Make install　　安装内核

4.使用新的内核启动，并用用户态程序验证

重启系统，按住 shift 进入 grub 选项



选择新的内核启动



编写用户态程序

```c
#include<stdio.h>
#include<unistd.h>
#include<sys/syscall.h>

int main()
{
    int temp;
    temp = syscall(314, 1);
    if(temp == 1)
        printf("success!\n");
    else
        printf("failed!\n");

    return 0;
}
```

使用 syscall 函数调用系统调用， 下图为运行结果。

```
gao@gao-virtual-machine:~$ gcc test.c
gao@gao-virtual-machine:~$ ./a.out
success!
```

观看系统日志 sudo dmesg

```
[   25.108646] cfg80211:   (5170000 KHz - 5250000 KHz @ 40000 KHz), (300 mBi, 20
00 mBm)
[   25.108647] cfg80211:   (5735000 KHz - 5835000 KHz @ 40000 KHz), (300 mBi, 20
00 mBm)
[   25.265885] e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: N
one
[   40.313399] audit_printk_skb: 36 callbacks suppressed
[   40.313402] type=1400 audit(1463890469.194:24): apparmor="DENIED" operation="
capable" parent=1629 profile="/usr/sbin/cupsd" pid=1649 comm="serial" pid=1649 c
omm="serial" capability=21  capname="sys_admin"
[  295.134346]           this is the project 1
[  295.134350]           hello myKernel!
gao@gao-virtual-machine:~$
```

在最后打印出了我这个简单的系统调用的内容，至此，说明实验取得了成功。

四、实验总结

此次实验是操作系统的第一个课程设计，主要学习到了系统调用的相关知识。

一开始有些摸不着头脑，主要是通过上网查找各种知识，在了解主要改动三处程序以后，就开始了对内核的修改，第一次经历三小时编译后因为程序里

有错误而编译失败，修改过后，并用了多核编译一个半小时编译成功，使用用户态程序验证也取得了成功。

总体上，此次实验虽然比较简单，但对于第一次接触修改内核源码的我来说，有点不知所措，仍然花了许多时间来研究，内核编译也很耗费时间，需要有极大的耐心才能完成此次实验。